Robustness Guarantees for Anonymity

Gilles Barthe*, Alejandro Hevia†, Zhengqin Luo‡, Tamara Rezk‡, Bogdan Warinschi§
*IMDEA Software - Madrid, Spain
Gilles.Barthe@imdea.org
†Universidad de Chile - Santiago, Chile
ahevia@dcc.uchile.cl
‡INRIA Sophia Antipolis-Méditerranée - Sophia Antipolis, France
{zluo,trezk}@sophia.inria.fr
§University of Bristol - Bristol, United Kingdom
bogdan@compsci.bristol.ac.uk

***Abstract*—Anonymous communication protocols must achieve two seemingly contradictory goals: *privacy* (informally, they must guarantee the anonymity of the parties that send/receive information), and *robustness* (informally, they must ensure that the messages are not tampered). However, the long line of research that defines and analyzes the security of such mechanisms focuses almost exclusively on the former property and ignores the latter.**

In this paper, we initiate a rigorous study of robustness properties for anonymity protocols. We identify and formally define, using the style of modern cryptography, two related but distinct flavors of robustness. Our definitions are general (*e.g.* they strictly generalize the few existent notions for particular protocols) and flexible (*e.g.* they can be easily adapted to purely combinatorial/probabilistic mechanisms).

We demonstrate the use of our definitions through the analysis of several anonymity mechanisms (Crowds, broadcast-based mix-nets, DC-nets, Tor). Notably, we analyze the robustness of a protocol by Golle and Juels for the dining cryptographers problem, identify a robustness-related weakness of the protocol, and propose and analyze a stronger version.

I. Introduction

A. Background and motivation

Privacy and robustness are essential properties of anonymous protocols. Together, anonymity and robustness ensure that protocols achieve their purported goal. Informally, privacy ensures that the identity of communicating parties is not revealed, whereas robustness ensures that messages reach their recipients uncorrupted. The following examples demonstrate the importance of robustness, and suggest why a quantitative analysis is highly desirable.

Cocaine auction (after [46]): Anonymous protocols can save your life. But do they allow you to do fair business in a ruthless market? The Cocaine Auction Protocol is a protocol designed to conduct anonymous electronic auctions when participants do not trust each other, and cannot rely on anonymous parties. When bidding for the last cocaine shipment, how can a bidder ensure that his bid will not be discarded by some competitor? Is it possible for a bidder to discard somebody else's bid?

Godfather election: Voting protocols permit to hold inconspicuous elections remotely. But do they protect established institutions from having their reputation entached by a fraudulous behavior? The Godfather Election Protocol is a variant of the Cocaine Auction Protocol where participants are to cast a single vote to elect the next Godfather—the election may have been triggered by an unfortunate flaw in a recent cocaine auction, making participants even less inclined to trust each other. How can a voter ensure that his vote will not be discarded by a member of a rival family, or even by his own cousin? And what is the penalty for discarding someone else's vote? If, for example, elections are held by majority voting, and the penalty for discarding somebody else's vote is to loose one own vote, then no voter will benefit from cheating.

There is a large body of literature on developing protocols that achieve the desired level of privacy (see [28] for an account of different anonymity notions), and on providing quantitative measures of privacy to evaluate the relative strength of a specific protocol or to compare the strengths of two protocols [43], [17]. In contrast, existing definitions of robustness are scarce, and may not enforce intuitive guarantees (see Section I-B). Moreover, a systematic study of the level of robustness provided by existing protocols is entirely missing. The goal of this work is to define, analyze, and compare different notions of robustness for anonymity protocols, using the tools of modern cryptography. The definitions provide precise definitions of robustness that guarantee that corrupt participants will only have limited impact on the set of exchanged messages. The definitions are global, i.e. they abstract away from the internal details of the protocols; more concretely, the definitions compare the set of messages that are sent by participants and the set of messages that reach their recipients. The global character of our definitions makes them intuitive, and applicable to a variety of settings. Our robustness notions also allow us to rigorously analyze (using provable cryptography) different protocols on the same bases, as for example mixnet-based anonymous channels where the underlying mixnet does

not provide strong correctness guarantees. Furthermore, our definitions fit well with *practical* anonymous channels (e.g. Tor [18]) where correctness guarantees do not hold with overwhelming probability. Finally, and most importantly, our definitions allow us to quantify the appropriateness of anonymous channels for cocaine auctions and Godfather elections.

B. A critical review of dining cryptographers

In some cases, the existent ad-hoc correctness notions may fail to achieve the expected guarantees. To explain this important point, we consider the short dining cryptographers protocol of Golle and Juels [26] for which an ad-hoc notion of robustness has been developed. We give here a high level description of the protocol, omitting technical issues that are not relevant to explain their notion of robustness.

The protocol heavily relies on zero knowledge proofs of knowledge (ZKPoK) [25]. In a nutshell, such proofs allow a prover (who possesses some piece of secret information) to convince a verifier of the veracity of a statement S, without revealing anymore information except for the validity of S. The protocol proceeds as follows.

1) Prior to running the protocol, all n participants are allocated a different slot between 1 and n;
2) Each participant emits a vector of n elements that contains special values except for the one position singled out by the slot allocation protocol. In this slot the participant places his "vote" multiplied by another special value. An adversary cannot see the difference between a special value and a vote multiplied by a special value. These special $n-1$ values satisfy properties (see short DC-nets in Section VI for details) that participants must prove using a ZKPoK proof. In addition, each participant must prove that he has at least $n-1$ special values in his vector (so to prevent participants to vote twice, for example);
3) In order to count votes, each participant validates the vectors of all other participants by using ZKPoK tests. All vectors that do not pass the tests are considered dishonest and are discarded. For each dishonest vector, a new valid vector, with a "blank vote", is generated, for example using threshold cryptography [38];
4) Once all vectors are validated, they are all multiplied. By properties of the special values, multiplication of all positions will "magically" cancel out, leaving a vector that contains the messages of the participants in clear.

Without the ZKPoK proofs, the correctness guarantees of the protocol would be very low, as a corrupt participant could garble the messages of others by emitting a vector that contains trash on all positions except the one provided to him by the slot allocation protocol. Thus, the ZKPoK tests are essential to guarantee the correctness of the protocol. Indeed, Golle and Juels [26] define correctness of DC-nets in terms of the (in-) ability of a corrupted participant to generate an invalid vector that will pass the ZKPoK tests. This definition of correctness is tied to the protocol phase in which the ZKPoK tests are performed, and it follows directly from the definition of ZKPoK schemes. One shortcoming of this definition of correctness is that it is intimately tied to the implementation of the cryptographic mechanisms used by a protocol, and that it does not provide end users abstract, protocol independent, guarantees.

Giving the correctness property of the protocol, a natural question then is: what does it imply? What are the robustness guarantees for cocaine auctioneers and Godfather electors? Does the correctness property entail that every bid for the cocaine shipment will be considered, or that every vote for the Godfather will be counted as expressed in the ballot? The answer is **No**. In fact, the above definition of correctness does not rule out a dishonest participant from emitting a valid vector that passes all ZKPoK tests and still prevents from accounting some bids or votes.

Assume that Gino has convinced Vito to vote for him, and has offered himself a place in his management team, if he is elected as Godfather. During a heated discussion at a family celebration, Vito's cousin Fredo made clear his intentions to vote for Toni. On the following day, Vito is able to find out Fredo's slot. Then, he produces a valid vector (with correct special values) with a message "Gino/Toni" and forces his message to the same position used by Fredo to vote for Toni. Then the multiplication after canceling out the random values results in a vote for Gino and the cancellation of a vote for Toni. One may wonder whether we can still claim that the DC-nets protocol is robust? It is certainly not robust for the guarantees that the organizers of the Godfather election were expecting from the protocol. In this paper, we propose a stronger version of the protocol that is fit for being used in Godfather's election.

C. Contributions

We summarize the main contributions of the paper.

Rigorous definitions: While the notion of robustness had been considered elsewhere in the particular contexts of various anonymity mechanisms, the individual occurrences use ad-hoc interpretations for robustness. This is not surprising since no general definition exists. The first contribution of this paper is a formalization of two flavours of robustness: communication and interference robustness. Our definitions follow a rigorous approach popular in modern cryptography. We define the two security notions through games between an adversary and the anonymity mechanism. Each game fixes the abilities of the adversary and specifies and quantifies what does it mean for the adversary to break the robustness of the protocol.

For each of the two notions we give the definitions in the asymptotic setting (where the advantage of the adversary is bounded by a negligible function of the security parameter).

Importantly, our definitions immediately yield concrete versions (where the success of the adversary is measured as a function of its running time). Moreover, we propose non-asymptotic definitions which are suitable for the analysis of anonymity mechanisms based on combinatorial and purely probabilistic properties, and not only protocols that rely on cryptographic primitives.

A crucial feature of our definitions is that they are independent of the particular network topology and communication model. Indeed, our definitions of robustness are expressed only in terms of the input/output relation of the protocol, and thus are meaningful without having to specify the particular network topology and communication model. Obviously, this is not to say that the robustness of particular anonymity mechanisms can be analyzed independently of the underlying infrastructure: indeed, these details need to be spelled out and taken into account when analyzing the robustness of particular anonymity mechanisms.

Analysis of existing mechanisms: We demonstrate that our definitions are very general through several examples where we quantify the robustness of existent anonymity mechanisms. Importantly, the particular protocols are quite different and are intended for different scenarios. Specifically, we consider Tor [18], Crowds [41], a broadcast protocols using mix-nets ([37], [29], [22], [48]), and DC-nets [26]. Figure I-C summarizes the robustness of the above anonymity protocols;

Strengthening the Golle-Juels protocol: Our analysis shows that the Golle-Juels short DC-nets protocol does not match our optimal interference robustness bound. Intuitively, this is because the protocol does not provide the honest parties with a mechanism to detect if the message they receive had been tampered with or not.

We therefore propose a strengthening of the protocol where we tag the messages that are being sent with a randomized tag: as long as tag contains enough min-entropy an adversary cannot tamper with the message without being detected. We prove that the resulting protocol has optimal interference robustness.

Contents of the paper: Section III briefly describes the setting studied in the paper. Section IV-A gives a high-level overview of the robustness properties that we propose and study in this paper and Section IV-B presents their formal definitions. Section VI compares and analyzes the robustness of Crowds, a protocol based on mixnets, and short DC-nets. Section V-B proposes a modification to the DC-net protocol that ensures that the interference robustness of the protocol is optimal. Finally, we conclude in Section VII.

II. Related work

The study of reliable communication in networks has a rich and extensive literature, from network coding (e.g. everything following [44]) to the study of algorithms for achieving delivery of messages over different network topologies under the presence of adversaries (e.g. [19]). Reliability may take more sophisticated forms as in broadcast channels whose main security property is consistency: the messages received by all players as a result of a broadcast transmission operation are guaranteed to be the same. The problem of achieving consistency when implementing broadcast on top of a point-to-point network (commonly known as Byzantine agreement) has received enormous attention (e.g. [34], [20], [11], [10] among many others).

In the context of anonymous communication, after Chaum's seminal paper [13], several attacks targeting mixnets' correct operation [40], [16] motivated researchers to find mechanisms to achieve more reliable operation although often with informal and ad-hoc definitions of robustness. (The terms *correctness* – meaning the output, if exists, is consistent with the inputs – and *robustness* – meaning correct output always exists – are often used interchangeably in the literature.) For instance, Ogata et al. [36] use threshold cryptography to minimize the effect of misbehaving mixers during the execution of a mixnet protocol. In terms of definitions, several previous works proposed intuitive formulations of robustness on various anonymous communication protocols: *correctness* for DC-nets [26]; *reliability* for Tor [8]; *robustness* for Mix-nets [29]. Some of these definitions (e.g. correctness) reflect informal understanding of what robustness could mean. However a rigorous definition is still missing, making it difficult to compare robustness guarantees for different protocols. Actually, their robustness results can be restated in our definitional framework, thus a unified comparison is possible (the proof of Theorem 6.5 borrows the result from the original paper [29]). Other definitions (e.g. reliability) are established on certain assumptions of adversarial strategies, which are less convincing since a different adversary might easily break the property.

Jakobsson, Juels and Rivest [30] and Boneh and Golle [7] consider relaxations to mixnet correctness so mixnets are allowed to fail with small but more than negligible probability. Both works, however, specifically focus on mixnet correctness so their definitions are not necessarily extensible to other (combinatorial and probabilistic) anonymous channel schemes. Yet, their constructions can indeed be used to implement anonymous channels – which can be rigorously analyzed under our definitions. A stronger approach in definitional terms was taken by Wikström who uses a simulation-based characterization (see discussion below) to define correctness as well as anonymity of mixnets [48]. Wikström's definition is cast in the Universally Composable framework which inherently captures the correctness requirement of any problem defined in this model but whose realization requires a highly sophisticated and rather complex protocol. An interesting question we leave for future work is whether our characterization of robustness can be paired with a reasonable privacy requirement (e.g. [28]) in order to imply a reasonable simulation-based formulation of

	k Communication Robustness	τ Interference Robustness	Remarks
Tor	-	-	Robustness is always broken non-negligibly
Tor	$(\alpha\mu, \epsilon)$	-	Only non-asymptotic comm-robustness holds
Crowds	-	-	Robustness is always broken non-negligibly
Crowds	$(\alpha\mu, \epsilon)$	$((t+1)n - \alpha\mu, \epsilon)$	Non-asymptotic robustness
Mix Networks [37]	$n - t$	$2t$	Authenticated broadcast, HM of mixers
Mix Networks [29]	$n - t$	$2t$	Authenticated broadcast, HM of mixers
GJ DC-Nets	$n - 2t$	$3t$	Authenticated broadcast, HM of senders
SGJ DC-Nets	$n - 2t$	$2t$	Auth. simultaneous broadcast, HM of senders

Figure 1. Summary of robustness properties for several anonymous channels protocols. In all cases n is the number of participants and t is the number of corrupt participants. For both Tor and Crowds we analyze asymptotic and non-asymptotic versions (the two entries in the table for each protocol correspond to these two settings). Here μ are different specific constant for both Tor and Crowds (details in Section VI), $\alpha \in (0, 1]$, and $\epsilon = e^{-\mu\alpha^2/2}$. HM stands for honest majority.

reliable anonymity.

On the practical side, there has been attempts to design anonymous channel protocols specially with reliability in mind (under some adversary model), concretely Cashmere [50] and Hydra-Onions [32]. Their definitions of reliability are rather ad-hoc, although they reflect some informal understanding of what robustness could mean. In fact, as Borisov et al. point out [8], the robustness of these protocols are still subject to attacks if the definition of robustness under which these protocols are supposed to work do not adequately capture some realistic adversarial strategies. As a consequence, developing formal and rigorous definitions of robustness that are general in nature seems essential to be able to compare robustness guarantees across anonymity protocols with seemingly different characteristics.

Our definitions use the game-based approach where security is defined through a game between the adversary and the primitive (see [4], [45] although the idea of indistinguishability games is much older [24]).

III. Protocols, Messages, and Adversaries

In this section we specify the setting that we investigate and fix some of the details common to all of our models and definitions. For any integer $s > 0$, let $[s]$ denote the set $\{1, \ldots, s\}$. We consider a setting with $n + 1$ parties (n is some fixed constant). Sender parties $P_1, P_2, \ldots, P_n$ attempt to send, respectively, messages $M_1, M_2, \ldots, M_n$ to a receiver party R, in an anonymous and reliable way. The communication between parties can be observed and even be influenced by an adversary A. However, we do not fix a particular model of interaction between the adversary and the parties. Indeed, while the viable strategies of the adversary are tightly linked to the communication model, its goals (both breaking anonymity and perturbing the communication) can be specified only in terms of the messages input to the protocol and the messages received by R. If $\vec{M}$ is a tuple of n messages for the sender parties we write $\mathsf{Exec}(P(\vec{M}), R, A)(\eta)$ for the output of party R when the parties $P = (P_1, P_2, \ldots, P_n)$ and R execute the protocol in the presence of adversary A for security parameter η. We assume that this output is a multiset[1] of messages, and we write $\vec{S} \stackrel{\$}{\leftarrow} \mathsf{Exec}(P(\vec{M}), R, A)(\eta)$ for assigning the output of R to multiset $\vec{S}$. Notice that the execution is randomized, so once $\vec{M}$ is fixed, $\mathsf{Exec}(P(\vec{M}), R, A)(\eta)$ is a random variable. Here, and throughout the paper we assume that the adversary corrupts statically at most t senders. Also, the definition assumes that any initialization that the protocol requires (i.e. a public key infrastructure) are performed at the beginning of the execution. Unless otherwise specified, this initialization and the cryptographic primitives use fresh random coins.

Our approach for avoiding to hardwire particular execution models in the definition is rather general and allows easy instantiations of our security definitions to the different possible settings. Relevant examples include passive eavesdroppers, asynchronous adversarial networks, broadcast and simultaneous broadcast networks with or without rushing adversaries.

Nevertheless, all our definitions can be made completely precise by requiring that protocols and the adversary are specified using a fixed language. An ideal language for this purpose is the language pWHILE (see e.g. [3], [21]). This language is a rather standard WHILE language extended with random sampling and procedure calls. One important advantage of using pWHILE is that the language has a fully formal semantics, together with a characterisation of probabilistic polynomial time operations, and that it can be used in machine aided proofs. In the setting of a pWHILE language, parties (honest and adversarial) communicate with each other through a shared memory. Corruption of parties as well as various flavors of secure channels are modeled by restricting the read/write access of the adversaries to the various memory locations in use. Specific syntactic restrictions can be used to smoothly account for the different communication media that are standard in cryptography. Two limitations of the language (with no consequence on the

[1] Robustness requires to preserve not only the set of delivered messages, but also their multiplicities. Strictly speaking, we are therefore concerned with multisets rather than sets. However, we take the freedom to refer to sets and vectors in our informal explanations.

results of this paper) are that the number of parties involved in the execution of a protocol needs to be fixed in advanced, and that the corruption model is static.

IV. Definitions of Robustness

A. Properties Overview

The correctness of an anonymity protocol can be measured by comparing the initial messages with the output messages. Consider a run of the protocol $\vec{S} \xleftarrow{\$} \mathsf{Exec}(P(\vec{M}), R, A)$. Note that some initial messages may not appear in the set of delivered messages, as a result of messages being lost, intercepted, or tampered. Conversely, some of the messages delivered may not be in the set of original messages, as they may have been forged by corrupt participants or may be the result of tampering some message that was in the initial set (think for example of a corrupt participant who garbles all messages in his power). In the worst case where no appropriate measure is taken to guarantee robustness, corrupt participants may be able to force that all honest participants may be prevented to publish their messages, and all published messages originate from corrupt participants or have been tampered. This suggests two measures for robustness: the first one, which we call communication robustness, simply measures the number of messages from honest participants that are found in the result of the protocol, i.e. in the final vector of messages. The second one, which we call interference robustness, quantifies the ability of corrupt participants to prevent honest participants from publishing their messages while publishing themselves their own messages, or more generally inducing the publication of spurious messages. Informally, interference robustness intends to capture the intuition that in some protocols, like those based on DC-nets [13], [26], corrupt participants must choose between publishing their message, or tampering (hence in some cases invalidating) a message by a honest participant. On the one hand, the communication robustness of the protocol is measured by $|\vec{M} \cap \vec{S}|$. Interference robustness of the protocol, on the other hand, is measured by $|\vec{M} \Delta \vec{S}|$ where Δ denotes the symmetric difference between two multisets.

B. Formal Definitions

Anonymity for the sending parties seems to imply that some integrity mechanisms like digital signatures and MACs cannot be used to ensure the integrity of messages that parties sent. It thus may seem that protocols for anonymity come with the undesirable property that the adversary can tamper, potentially undetected, with the messages sent by the honest parties. We discuss two related yet distinct threats posed by adversaries against anonymity mechanisms.

Communication robustness is concerned with the ability of an adversary to block the messages sent by an honest party. In other words, communication robustness measures how many of the messages sent by honest parties are received by the receiver. Clearly, a fully active adversary can easily achieve his goal by simply dropping all of the messages sent by honest parties so this case is not particularly interesting and the notion may not even make sense. However, in media where there are some guarantees for communication, for example broadcast channels or the existence of direct channels between the honest participants and the destination, the notion makes sense and deserves to be investigated.

To define this notion we consider a game where an adversary A selects a vector of n messages $\vec{M}$ to be sent by the parties involved in the protocol (these parties include those under adversarial control) and then the adversary engages in the execution of the protocol P with parties $P_1, P_2, \ldots, P_n$ and receiver R. We say that protocol P is k-communication robust if the messages received by R contain at least k of the messages in $\vec{M}$. This intuition is captured by the following formal definition.

*Definition 1 (k-*Communication Robustness*):* We define the advantage of an adversary A against the k- communication robustness of protocol P by $\mathbf{Adv}^{k\text{-crob}}_{P,A}(\eta) =$

$$\Pr\left[|\vec{S} \cap \vec{M}| < k \mid \vec{M} \xleftarrow{\$} A;\ \vec{S} \xleftarrow{\$} \mathsf{Exec}(P(\vec{M}), R, A)(\eta)\right]$$

We say that the protocol P is k-communication robust (for a given execution model) if the advantage of any probabilistic polynomial time adversary A is negligible as a function of the security parameter η.

In the definition, we take a conservative approach in that we let the adversary choose the messages. Weaker variants where the message distribution is fixed but unknown to the adversary are possible and straightforward. (Our choice is driven by the wider applicability of the current definition and the fact our definition can already be satisfied by known protocols.)

In the above $\vec{S} \cap \vec{M}$ is standard multiset intersection (even if $\vec{M}$ is an ordered set of messages). The definition above is suitable for systems where security properties follow from the security of underlying cryptographic primitives. Yet, many existent systems intended to provide anonymity guarantees are based on other probabilistic mechanisms. Our definition of robustness can be easily adapted for this case: we drop the security parameter and require that the probability that the number of messages in $\vec{S} \cap \vec{M}$ is less than k is bounded.

*Definition 2 ((k,ϵ)-*Communication Robustness*):* We say that protocol P is (k,ϵ)-communication robust if $\mathbf{Adv}^{k\text{-crob}}_{P,A} =$

$$\Pr\left[|\vec{S} \cap \vec{M}| < k \mid \vec{M} \xleftarrow{\$} A;\ \vec{S} \xleftarrow{\$} \mathsf{Exec}(P(\vec{M}), R, A)\right] \leq \epsilon$$

Security in the sense of communication robustness does not really clarify *how* the adversary tampers with the messages that are not sent. Of course, corrupt senders can always change their input message and this can not really be prevented. However, it should not be the case that the

adversary can change (undetected) some of the messages of the honest senders and replace them with their own. We thus propose the notion of *interference robustness* which quantifies precisely the ability of the adversary to have the receiver accept messages that are not part of those input to the protocol at the expense of messages that are. The following definition says that the goal of an adversary against interference robustness is to maximize the number of honest messages that are not delivered together and the number of messages that were delivered but were not input to the protocol.

*Definition 3 (k-*interference Robustness*):* We define the advantage of an adversary A against the k-interference robustness of protocol P by $\mathbf{Adv}^{k\text{-nrob}}_{P,A}(\eta) =$

$$\Pr\left[|\vec{S}\Delta\vec{M}| > k \mid \vec{M} \xleftarrow{\$} A;\ \vec{S} \xleftarrow{\$} \mathsf{Exec}(P(\vec{M}), R, A)(\eta)\right]$$

We say that protocol P is k-interference robust if the advantage of any probabilistic polynomial time adversary A is negligible as a function of η.

As for communication robustness, we can define a non-asymptotic version of this security notion.

*Definition 4 ((k,ϵ)-*interference Robustness*):* We say that protocol P is (k,ϵ)-interference robust if $\mathbf{Adv}^{k\text{-nrob}}_{P,A} =$

$$\Pr\left[|\vec{S}\Delta\vec{M}| > k \mid \vec{M} \xleftarrow{\$} A;\ \vec{S} \xleftarrow{\$} \mathsf{Exec}(P(\vec{M}), R, A)\right] \leq \epsilon$$

In a setting with n senders out of which t are corrupt it is unavoidable that t of the messages received by the receiver were not part of the input of the senders; corrupt parties can always change their input message and then run the protocol honestly. This shows that the constant k in the definition of communication robustness is always between 0 and $n-t$. Similarly, the above example shows that k in the definition of interference robustness is at least $2t$, in which case all corrupted participants replaced their messages. In the worst case possible where the adversary replaces all of the n input messages with different ones interference robustness can reach $2n$ (the worst case depends on how many messages can be accepted by the receiver). In the next section we show that the $n-t$ upper bound for communication robustness and the $2t$ lower bound for interference robustness are optimal (in that they can be attained).

V. Analysis and Improvement of DC-nets

A. DC-nets robustness

The DC-net protocol proposed by Golle and Juels [26] directly extends the ideas from Chaum's DC-net [13]. In Chaum's protocol, parties share secret keys (called *keypads*) in a pairwise fashion which satisfies that the addition of all keypads effectively cancels out. To broadcast a message a party P_i adds (`xors`) the message into one of her pairwise-shared keypads so when keys are later combined, keys cancel out leaving her message as the result. Golle and Juels' protocol dispense with sharing private keys by having each pair of parties non-interactively compute the keypads as Diffie-Hellman keys. These keypads satisfy the same property – they cancel out once all are combined - and enjoy specific algebraic properties (inherited from the pairing-based key agreement used) that allow public verification of correctness of each party's keypad. In consequence, mis-constructed keypads can be identified. Moreover, since each party's private keys is initially threshold-shared among for all parties, reconstruction of incorrectly-computed keypads is possible. Details follow.

Let p, n be integers, $e\colon G_1 \times G_2 \to G$ be an admissible bilinear map over finite groups G_1, G_2, G, where $|G_1| = |G_2| = p$, and let g, h be generators of G. Golle and Juels' protocol (short DC-Nets) is composed of three phases. In the first phase, each party P_i gets a private key $x_i \in \mathbb{Z}_p$ and a public key $y_i = g^{x_i}$, while each other party P_j gets a (n,t)-share $x_{i,j}$ of x_i. In the second phase, in order to send message M_i, party P_i first chooses a random $c_i \in [n]$, and then creates a n-dimensional vector $W_i = (W_{i,\ell})_{\ell\in[n]}$, where each keypad $W_{i,\ell}$ is the multiplication of $e(q_\ell, y_j)^{x'_i}$ over all $j \neq i$, where x'_i is carefully chosen as x_i (if $i < j$) and $-x_i$ if not; q_ℓ is a hash of the concatenation of ℓ with a session id. Party P_i concludes the keypad computation by multiplying M_i into the c_i-th keypad so it becomes $M_i \cdot W_{i,c_i}$ (clearly, if P_i is the only sender, this choice of parameters produces $\Pi_i W_{i,\ell} = 1$ for all $\ell \neq c_i$ and $\Pi_i W_{i,c_i} = M_i$). Party P_i then broadcast her vector W_i together with a non-interactive zero-knowledge proof [42] that W_i was correctly computed. A vector of a participant is proved as correctly computed if it contains correct keypads in every position except for one. Besides this, there is a (zero knowledge) proof of knowledge that shows that the participant knows which is the exceptional position. In case a zero-knowledge proof fails for some malicious party's keypad (say that of P_k), a threshold of t parties recover P_k's private keys x_k from their shares and publicly reconstruct P_k's keypads. Messages for each $\ell \in [n]$ are obtained by simply pointwise multiplying all broadcast vectors.

Informal analysis of the protocol: Assuming that the honest participants follow a slot reservation protocol to avoid collisions between positions where they transmit messages, there will be at most t collisions after a run of a protocol. These collisions are produced by corrupted participants who do not follow the slot allocation protocol. This already implies that there is at least an adversary that prevents t honest messages to be correctly delivered, and hence the protocol is $(n-2t)$-communication robust. If an honest vector W_H and a dishonest vector W_D transmit messages m_H and m_D in the same position, none of the m_H or m_D can be recovered (the keypads will not cancel out) as such, but instead a message $m_H * m_D$ will be delivered as a correct message. Using this strategy, the adversary not only tampers message m_H but also, from the viewpoint of the receiver, he cannot be detected. Hence, he interfere with the set of

received honest messages.

We actually show that short DC-nets is n-2t-communication robust and 3t-interference robust.

Theorem 5.1: Short DC-Nets is $n - 2t$-communication and $3t$-interference robust.

A proof of the above theorem, using a fixed execution model for the protocol that is specified in pWHILE, can be found in Appendix C. A full description of the protocol in pWHILE and cryptographic assumptions used in the proof for the non-interactive proof of knowledge schemes can be found in Appendix A and B.

B. Strong Golle-Juels: improving interference robustness

One weakness of the Golle-Juels protocol (Section V-A) is that a malicious sender can tamper with the message sent by an honest party in undetectable way: the adversary can simply place a value c in the slot allocated to some honest party P_i. The message that is then received on this slot is $m_i \cdot c$. In this section we propose a way of improving the interference robustness of the protocol. Essentially, we append to the message a randomized tag such that the above attack can be detected. The scheme that we propose could be useful in other contexts and we describe it next.

C. Obliviously non-malleable randomized tagging schemes

We define a randomized tagging scheme as a scheme $(\mathsf{tag}, \mathsf{ver})$ given by a randomized algorithm tag for tagging and a deterministic verification algorithm ver. It is required that if $t \xleftarrow{\$} \mathsf{tag}(m)$ then $\mathsf{ver}(m, t) = 1$. Informally, the desired security property from a tagging scheme is a form of non-malleability. An adversary that can tamper with a message-tag pair (m, t) in some predefined ways cannot create a new valid message-tag pair (m', t') without knowing the (freshly computed) tag t. To present the notion that we need for this paper we introduce a bit of notation. We write $\langle m, t\rangle$ for a reversible encoding of a message-tag pair as a group element in some group $(G, \cdot)$. By abuse of notation, for a group element g we write $\mathsf{ver}(g)$ for the result of decoding g as a pair (m, t) and then applying the verification algorithm ver to this pair.

We then require that the quantity:

$$\Pr\left[\mathsf{ver}(c \cdot \langle m, t\rangle) = 1 \mid (m, c) \xleftarrow{\$} A;\ t \xleftarrow{\$} \mathsf{tag}(m)\right]$$

is negligible in the security parameter, for any probabilistic polynomial time adversary A. The probability is over the coins of the adversary and over the coins used in tag and in the above we require $c \neq 1$ (since otherwise the adversary would trivially win). Informally, a tagging scheme that satisfies the above definitions ensures that an adversary who is only allowed to multiply a value c to an encoding of some message m together with a randomized tag t, only succeeds in producing a new valid encoding $\langle m', t'\rangle$ with negligible probability.

Constructions: A simple construction of a tagging scheme as above uses a hash function H and is as follows. To tag a message m, the algorithm tag selects a random string r of length η (η is some security parameter) and returns $r||H(m||r)$. To verify that a tag $c = r||h$ is valid for message m the ver algorithm checks that $h = H(r||m)$ and accepts if this is the case, and rejects otherwise. It can be easily shown that in the random oracle model that the tagging scheme defined above is secure according to our definition. An alternative construction could possibly be based on the non-malleable functions recently introduced by Boldyreva et al [6].

D. Our strengthening of the Golle-Juels protocol

Our version of the Golle and Juels protocol replaces the messages sent by parties with their tagged versions.

The transformation affects the phases of transmission when messages are multiplied with the padding. In the transmission phase now messages are concatenated with a fresh value x and a hash of the message concatenated with the fresh value:

$$v_i[\ell] := (M_i||x||H(M_i||x)) \cdot w_i[\ell]$$

where $v_i[\ell]$ is the vector position where the message M_i should be transmitted. The message is concatenated with x and a tag and then multiplied by the padding of the protocol. The transformation also requires that the transmission and reconstruction phases are implemented by simultaneous broadcasting [27] (in contrast to the original protocol where we only need broadcasting). The property is that if the adversary does not know the fresh value x before receiving all messages, then only with negligible probability he will be able to generate by collisions a message satisfying the expected format.

Once all vectors are considered valid, and multiplication of all vectors is done, there is a further phase where validity of messages is verified. If a message does not fit with the format described above, then the message is dropped as invalid (in our definition of protocol, this means that the message is not included in multiset S).

Theorem 5.2: The Strong Golle-Juels protocol is $n - 2t$-communication robust and $2t$-interference robust.

VI. Analysis of Tor, Crowds, and Mix-nets

A. Tor robustness

Tor implements *The Onion Routing* protocol to prevent an end-point user from being traffic analyzed [18]. A Tor network contains a number of *Onion Routers*. An end-point user anonymizes its communication with certain destinations by building a path among these routers. Along the path, each router is only aware of its predecessor and successor, therefore the origination of the communication is protected under certain limited local passive adversary. We present the

essential of the Tor implementation and an adversarial model for the analysis of its robustness.

A Tor network contains the following components:

1) m Onion Routers: $OR_1, OR_2, \ldots, OR_m$;
2) Each router owns a pair of keys (PK_1, SK_1) for public-key encryption. PK_1 is public and SK_1 is private;
3) The communication channel between each router and between routers and originators is encrypted via TLS (Transport Layer Security Protocol).

A communication over Tor network can be divided into two phases: the path (circuit) setup phase and the message transmission phase.

1) On the setup phase, the originator, say Alice, will first create a path containing a single router. Alice selects one of the router OR_{i1} (possibly random), sends OR_{i1} a $\{CREATE\}$ message, and uses OR_{i1}'s public key to exchange a fresh symmetric encryption key K_{i1} between Alice and OR_{i1}. Meanwhile, the router will create a fresh ID to record this path;
2) Suppose now Alice has established a path with k routers $OR_{i1}, \ldots, OR_{ik}$, and it shares symmetric encryption keys separately with each router. It now can extend the path to $k+1$ routers by sending a relay message $\{EXTEND, OR_{i(k+1)}\}$ to the last router OR_{ik} in the path. This message will be encrypted with keys shared by Alice and each router in a reverse order according to the path, like an onion. Each router will rip off one layer of encryption when the message is being relayed. Thus only the last router OR_{ik} in the path can actually see the content of the message and extends the path to $OR_{i(k+1)}$;
3) Finally, when the path is ready, Alice can relay any message to a receiver through the path. Since each relayed messages will be encrypted and decrypted along the path, only the last router can see the real destination of the message;

We also make some reasonable assumptions for proving robustness in this setting:

1) There are n users. Each user wants to send one message to a given destination R;
2) There is a fix number of onion routers in the network, and p ($0 \leq p \leq 1$) is the fraction of corrupted routers controlled by the adversary;
3) Each user selects new router by uniformly random choice, and each path will contain l routers;
4) The adversary fully controls corrupted routers and corrupted users;

An obvious observation is that there must be no corrupted router in the path, in order to faithfully deliver the originator's message.

Proposition 6.1: Assume that there are n users for the Tor network, in which $n-t$ are honest and t are corrupted. It does not satisfy k-communication robustness for any $k > 0$; and it does not satisfy τ-interference robustness and (τ, ϵ)-interference robustness for any $\tau > 0$ and $\epsilon > 0$.

Proof: Since the fraction of corrupted routers is fixed, there exists a constant probability c such that there is at least one corrupted router in every path established by honest users. Those malicious routers can choose to drop any traffic through them, by which we can break the cryptographic communication robustness with a constant probability.

For the interference robustness and its non-asymptotic version, since corrupted users and corrupted routers can collude to reveal the destination R, and fill R with unlimited number of dummy messages. Therefore for any $\tau > 0$, we can break the interference robustness with constant probability 1. ■

We can show a parameterized bound for the non-asymptotic communication robustness.

Theorem 6.1: Assume that there are n users for the Tor network, in which $n-t$ are honest and t are corrupted and let $\phi = (1-p)^l$. Then for any $\alpha \in (0, 1]$, the Tor network is $(\alpha\mu, \epsilon)$-communication robust, where $\mu = \phi(n-t)$ and $\epsilon = e^{-\mu\alpha^2/2}$.

Proof: We give a proof sketch here. We can observe that if a path does not contain a corrupted router, then the message is guaranteed to be delivered. The probability ϕ that a path does not contain a corrupted router is

$$\phi = (1-p)^l$$

Let X_i be the random variable denoting number of messages delivered for the i-th honest participant. Notice that at least a corrupted router occurs in the path with probability $1-\phi$, then X_i equals to 0, and with probability ϕ, X_i equals to 1. Let $X = \sum_{i=1}^{n-t} X_i$ be the random variable denoting number of messages delivered for all honest participants.

We take the expectation value of X as the number of honest messages that is expected to be delivered in Tor. Since $X_1 \ldots X_{n-t}$ are independent (because each choice of either forwarding or delivering is independent for honest routers), we have

$$\mu = E(X) = E\left(\sum_{i=1}^{n-t} X_i\right) = \sum_{i=1}^{n-t} E(X_i) = \phi(n-t)$$

To prove (k, ϵ)-communication robustness for some k and ϵ, we need to bound the probability $\mathbf{Adv}_{P,A}^{k\text{-crob}} =$

$$\Pr\left[|\vec{S} \cap \vec{M}| < k \mid \vec{M} \xleftarrow{\$} A; \vec{S} \xleftarrow{\$} \mathsf{Exec}(P(\vec{M}), R, A)\right]$$

to ϵ. Observe that random variable X is exactly equal to $|\vec{S} \cap \vec{M}|$, then we can apply Chernoff bound to obtain a parametric bound. Assuming that $\alpha \in (0, 1]$, we can show that

$$\Pr[X < \alpha\mu] < e^{-\mu\alpha^2/2}$$

Therefore Tor is $(\alpha\mu, e^{-\mu\alpha^2/2})$-communication robust.

■

As we mentioned before, Borisov et al.[8] have (informally) proposed reliability for anonymous communication which is similar to our definition of communication robustness. They have analyzed the reliability for Tor and Mixnet. However their definition for reliability is ad-hoc for different protocols and different adversarial strategy, whereas our definition is independent from protocols and adversaries being analyzed. For example, the reliability for Tor is based on a specific type of selective Denial-of-Service attack. The adversary who controls a number of routers will refuse to provide relay service if it does not control the first and the last node of the path. The Tor network is reliable (or usable) only when there is no corrupted router in the path, or the first and last node of the path are both corrupted. Reliability does not imply communication robustness in general, since different adversarial strategies (e.g. corrupted routers simply dropping any messages passing through it) yields different interpretations of reliability.

B. Crowds robustness

Basically, a Crowd is just a group of users which collectively send messages to a destination not in the group in order to achieve sender anonymity. Given a particular Crowd, we assume that there are n users (called *jondos*) in the crowd; from those, t are corrupted. It is required that each pair of jondos is connected in the network, and share a key to encrypt traffic between them. A honest jondo behaves as follows when initiating a new message or receiving a forwarded message.

- With probability p_f, it selects uniformly at random one jondo from all users except the destination, and forwards the message;
- With probability $1 - p_f$, it submits the message to the destination, a receiver typically outside the Crowd.

We consider a slightly variant of the protocol. First, all users are assumed to send exactly one message.

Second, since the setting of this paper applies to sender anonymous protocols (although, we conjecture its extension to other kinds of anonymity properties is possible), we assume that the destination of all messages is a receiver R. Thus, at the end of an honest run of the protocol, R should receive n messages where $n - t$ are the messages from honest participants.

Third, we assume that R stops receiving messages from a particular participant after R has received n messages from that participant.

In order to disrupt the protocol, the adversary can choose to let a corrupted jondo to drop messages that it receives instead of forwarding them as specified for an honest participant (preventing e.g. that all messages passing by that jondo be delivered). Alternatevely, he can send more than one (dummy) message to R.

Proposition 6.2: Assume that there are n participant for the Crowds protocol, in which $n - t$ are honest and t are corrupted. Crowds does not satisfy k-communication robustness for any $k > 0$; and it does not satisfy τ-interference robustness for any $\tau < (t + 1)n$.

Proof: Since the probability p_f of forwarding is not dependent on the security parameter, there is a constant probability c that for every path there exists a malicious jondos. Because malicious jondos can choose to drop any honest, therefore this constant probability c also account for no honest message is delivered, by which we can break the cryptographic communication robustness and interference robustness with a constant probability. ■

We show here a parameterized bound for the non-asymptotic communication robustness and interference robustness.

Theorem 6.2: Assume that there are n participant for the Crowds protocol, in which $n - t$ are honest and t are corrupted and let ρ be $\frac{n(1-p_f)}{n-p_f(n-t)}$. Then for any $\alpha \in (0, 1]$, the Crowds protocol is $(\alpha\mu, \epsilon)$-communication robust, where $\mu = \rho(n - t)$, and $\epsilon = e^{-\mu\alpha^2/2}$.

Proof idea. The proof of this theorem is similar to the proof of Theorem 6.1. We can observe that if a path (initiated by a honest jondo) does not contain a corrupted jondo, then the message is guaranteed to be delivered. Let p_l be the probability that a corrupted jondo does not occur at the first l positions in the path, but occurs at the $l+1$ position. Then

$$p_l = \left(p_f \cdot \frac{n-t}{n}\right)^{(l-1)} \cdot \left(p_f \cdot \frac{t}{n}\right)$$

where $(p_f \cdot \frac{n-t}{n})$ is the probability to forward to a honest jondo, and $(p_f \cdot \frac{t}{n})$ is the probability to forward to a corrupted jondo. Then the probability ρ that no corrupted jondo will occur in the path is

$$\rho = 1 - \sum_{i=1}^{\infty} p_i = \frac{n(1-p_f)}{n - p_f(n-t)}$$

And the choice of the path are independent for each user. Therefore Crowds is $(\alpha\mu, \epsilon)$-communication robust.

Theorem 6.3: Assume that there are n participant for the Crowds protocol, in which $n - t$ are honest and t are corrupted. Let ρ be $\frac{n(1-p_f)}{n-p_f(n-t)}$. Then for any $\alpha \in (0, 1]$, the Crowds protocol is $((t + 1)n - \alpha\mu, \epsilon)$-interference robust, where $\mu = \rho(n - t)$, and $\epsilon = e^{-\mu\alpha^2/2}$.

Proof: We can observe that the only ways that the corrupted participants can influence the interference are preventing honest messages from being delivered, and sending dummy messages to the receiver R. Therefore, in the worst case, the final set of message $\vec{S}$ received by R contains only dummy messages sent by corrupted participants and delivered by honest messages. We have that

$$|\vec{S}\Delta\vec{M}| = (|\vec{S}| - |\vec{S} \cap \vec{M}|) + (|\vec{M}| - |\vec{S} \cap \vec{M}|)$$

The quantity $(|\vec{S}| - |\vec{S} \cap \vec{M}|)$ is the number of dummy messages sent by corrupted participants. It is bounded by $t \cdot n$, since R only accepts n messages from each participants. Thus, we have

$$|\vec{S}\Delta\vec{M}| \quad = \quad t \cdot n + n - |\vec{S} \cap \vec{M}|$$

We can now apply the inequality we obtained from the proof of communication robustness of Crowds protocol, such that

$$\begin{aligned} & \Pr\left[|\vec{S}\Delta\vec{M}| > (t+1)n - \alpha\mu\right] \\ = \; & \Pr\left[(t+1)n - |\vec{S} \cap \vec{M}| > (t+1)n - \alpha\mu\right] \\ = \; & \Pr\left[|\vec{S} \cap \vec{M}| < \alpha\mu\right] \\ \leq \; & e^{-\mu\alpha^2/2} \end{aligned}$$

Therefore, the Crowds protocol satisfies $((t+1)n - \alpha\mu, \epsilon)$-interference robustness. ∎

C. Mix-nets robustness

A mix-net is a protocol in which messages (say, emails) traverse several routers (or mixers) and, in the process, are shuffled or "mixed" with other messages with the intention that the relation to the original sender be lost. Since Chaum's seminal paper [12], research in the area has been extensive, from concrete mix-net proposals (see [39], [1], [31], [35], [22] among many others) to very practical protocols based on mix-nets (e.g. [47], [33], [14], [18] and references therein).

Mix-nets can be used to provide anonymous broadcast channels as follows:

1) On the setup phase, all parties set up a threshold encryption scheme [15] with public key y_e.
2) To send a message M_i, each party P_i encrypts it under key y_e and broadcasts it. From now on, each party (sender) acts as a mixer. Once all parties have submitted their encrypted input, the first mixer starts the mix-net.
3) In sequence, each mixer takes a vector of encrypted values, permutates it according to a random permutation and re-encrypts them using key y_e again in order to unlink the input vector to the output vector of encrypted values. Each mixer publicly proves the correctness of her mix by producing a valid Zero-Knowledge Proof of Knowledge proof (e.g. using [22]), i.e. each mixer must exhibit a *proof of correct shuffle* to prove with overwhelming probability that no messages were added or modified.
4) The last mixer broadcasts a vector of encrypted values $C = (C_i)_{i \in [n]}$. Then, all parties threshold decrypt (as many times as number of mixers) each ciphertext C_i obtaining values M_i^*'s which are broadcast.

This protocol, which we call Mix-net-AC, was suggested by Part et al. [37]. It tolerates a minority of corrupted mixers. As we show below, optimal robustness can be achieved using the proofs by Park et al., although more efficient proofs of shuffle can be used, for example, those proposed by Neff [35] and Furukawa and Sako [22].

Theorem 6.4: Assuming the existence of authenticated channels and at least $n - t$ honest mixers (where $t < \frac{n}{2}$), Mix-net-AC satisfies $n-t$ communication robustness, where there are $n - t$ honest participants and t corrupted participants. Furthermore, under the same hypotheses, Mix-net-AC satisfies $2t$-interference robustness.

Notice that the claim of the theorem is correct independently of the network topology since we assume that participants use authenticated broadcast channels to send messages (see [5] for an example of such a protocol in the context of wireless networks.)

Proof idea. From the hypothesis on the existence of authenticated channels, we have that corrupted participants do not have the ability to influence the communication between honest participants and the servers. Hence at least $n - t$ messages are assured to be delivered. Furthermore, the soundness property of the Zero-Knowledge Proof of Knowledge proof [25] of correct mixing guarantees that dishonest mixers cannot tamper with honest messages except with negligible probability.

An alternative construction that achieves the same level of robustness was proposed by Jakobsson and Juels in [29], where a definition of robustness of Mix-nets is also given. Robustness in their definition requires that if only a small group of mixers is corrupted, then all the honest messages are guaranteed to be delivered. Actually we can restate this theorem with our definition of robustness.

Theorem 6.5: Assuming that only minority of mixers are corrupted, and there is $n - t$ honest participants and t corrupted participants, the mix-net protocol in [29] achieves $n-t$ communication robustness, and $2t$-interference robustness.

VII. Conclusion

We have proposed two notions of robustness that are applicable to the case of anonymity protocols: communication robustness and interference robustness. These notions, suitable for provable cryptography, are also defined in a quantitative way, suitable for proving purely probabilistic protocols such as Crowds. As a proof of concept, we have compared and shown correctness, for the different notions of robustness, for three well-established anonymity protocols: Crowds, short DC-nets, and a Mix-net-based protocol (Mix-net-AC). From the three protocols analyzed, Mix-net-AC has optimal robustness but under the assumption that a majority of participants remains honest. Under the same assumption, but improving in efficiency (less number of rounds than Mix-net-AC), short DC-nets, proposed by Golle

and Juels, is the protocol that excels next.[2] We have also defined the necessary conditions to obtain optimal interference robustness in short DC-nets (in the modified protocol, the first phase using a broadcasting protocol is replaced by simultaneous broadcasting, thus showing that optimal interference-robustness has a cost in terms of efficiency). One open question we also leave for future work is whether it is possible to exhibit a general construction transforming any protocol that has a weak interference robustness into a protocol that is optimally interference robust.

ACKNOWLEDGMENTS: This work was partially supported by STIC-AmSud FMCrypto Project, French ANR SESUR-012, SCALP, Spanish project TIN2009-14599 DESAFIOS 10, Madrid Regional project S2009TIC-1465 PROMETIDOS, CONICYT via Fondecyt 1070332, and eCrypt-2 Network of Excellence.

REFERENCES

[1] M. Abe. Universally verifiable mix-net with verification work indendent of the number of mix-servers. In *EUROCRYPT*, pages 437–447, 1998.

[2] M. Backes and D. Unruh. Computational soundness of symbolic zero-knowledge proofs against active attackers. In *IEEE Computer Security Foundations Symposium, CSF 2008*.

[3] G. Barthe, B. Grégoire, and S. Zanella. Formal certification of code-based cryptographic proofs. In *36th symposium Principles of Programming Languages*, 2009.

[4] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *EUROCRYPT 2006*. Springer, 2006.

[5] M. Blaze, J. Ioannidis, A. D. Keromytis, T. Malkin, and A. D. Rubin. War: Wireless anonymous routing. In B. Christianson, B. Crispo, J. A. Malcolm, and M. Roe, editors, *Security Protocols Workshop*, volume 3364 of *Lecture Notes in Computer Science*, pages 218–232. Springer, 2003.

[6] A. Boldyreva, D. Cash, M. Fischlin, and B. Warinschi. Foundations of non-malleable hash and one-way functions. In *ASIACRYPT*, pages 524–541, 2009.

[7] D. Boneh and P. Golle. Almost entirely correct mixing with applications to voting. In V. Atluri, editor, *ACM Conference on Computer and Communications Security*, pages 68–77. ACM, 2002.

[8] N. Borisov, G. Danezis, P. Mittal, and P. Tabriz. Denial of service or denial of security? In *Proceedings of the 14th ACM conference on Computer and communications security*, page 102. ACM, 2007.

[9] C. Cachin and J. Camenisch, editors. *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*. Springer, 2004.

[10] C. Cachin, K. Kursawe, F. Petzold, and V. Shoup. Secure and efficient asynchronous broadcast protocols. In *Advances in Cryptology – CRYPTO ' 2001*, volume 2139 of *LNCS*, pages 524–541. Springer-Verlag, 2001.

[2] In fact, both Mix-nets and DC-nets implicitly assume honest majority as they all require authenticated broadcast channels which – if not provided as a primitive – do require a majority of the communicating parties be honest.

[11] R. Canetti and T. Rabin. Fast asynchronous Byzantine agreement with optimal resilience (extended abstract). In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, pages 42–51, 1993.

[12] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–88, 1981.

[13] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *J. Cryptology*, 1(1):65–75, 1988.

[14] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *In Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 2–15, 2003.

[15] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In G. Brassard, editor, *Advances in Cryptology—CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 307–315. Springer-Verlag, 1990, 20–24 Aug. 1989.

[16] Y. Desmedt and K. Kurosawa. How to break a practical mix and design a new one. In B. Preneel, editor, *Advances in Cryptology — EUROCRYPT'2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 557–572. Springer–Verlag, 2000.

[17] C. Díaz, S. Seys, J. Claessens, and B. Preneel. Towards measuring anonymity. In *PET' 02*, 2002.

[18] R. Dingledine, N. Mathewson, and P. F. Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, pages 303–320. USENIX, 2004.

[19] C. Dwork, D. Peleg, and E. Upfal. Fault tolerance in networks of bounded degree. *SIAM J. Comput.*, 17(5):975–988, 1988.

[20] P. Feldman and S. Micali. Optimal algorithms for byzantine agreement. In *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing*, 1988.

[21] C. Fournet and T. Rezk. Cryptographically sound implementations for typed information-flow security. In *35th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'08)*, pages 323–335, San Francisco, USA, Jan. 2008. ACM Press.

[22] J. Furukawa and K. Sako. An efficient scheme for proving a shuffle. In J. Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 368–387. Springer, 2001.

[23] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. *J. Cryptol.*, 20(1):51–83, 2007.

[24] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Science*, 28:270–299, 1984.

[25] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[26] P. Golle and A. Juels. Dining cryptographers revisited. In Cachin and Camenisch [9], pages 456–473.

[27] A. Hevia. Universally composable simultaneous broadcast. In R. D. Prisco and M. Yung, editors, *Security and Cryptography for Networks, 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006, Proceedings*, volume 4116 of *Lecture Notes in Computer Science*, pages 18–33. Springer, 2006.

[28] A. Hevia and D. Micciancio. An indistinguishability-based characterization of anonymous channels. In N. Borisov and I. Goldberg, editors, *Privacy Enhancing Technologies Symposium - Proceedings of PETS'08*, volume 5134 of *Lecture Notes in Computer Science*, pages 24–43, Leuven, Belgium,

July 2008. Springer.
[29] M. Jakobsson and A. Juels. An optimally robust hybrid mix network. In *PODC '01: Proceedings of the twentieth annual ACM symposium on Principles of distributed computing*, pages 284–292, New York, NY, USA, 2001. ACM.
[30] M. Jakobsson, A. Juels, and R. L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX Security Symposium*. USENIX, 2002.
[31] M. Jakobsson, A. Juels, and R. L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *Proceedings of the 11th USENIX Security Symposium*, pages 339–353, Berkeley, CA, USA, 2002. USENIX Association.
[32] M. K. Jan Iwanik, Marek Klonowski. Duo–onions and hydra–onions – failure and adversary resistant onion protocols. In *IFIP TC-6 TC-11 Conference on Communications and Multimedia Security 2004*, pages 1–15. Springer Verlag, 2005.
[33] D. Kesdogan, J. Egner, and R. Büschkes. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *Proceedings of the Second International Workshop on Information Hiding*, pages 83–98, London, UK, 1998. Springer-Verlag.
[34] L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.
[35] C. A. Neff. A verifiable secret shuffle and its application to e-voting. In *ACM Conference on Computer and Communications Security*, pages 116–125, 2001.
[36] W. Ogata, K. Kurosawa, K. Sako, and K. Takatani. Fault tolerant anonymous channel. In Y. Han, T. Okamoto, and S. Qing, editors, *Information and communications security: first international conference, ICICS '97, Beijing, China, November 11–13, 1997: proceedings*, volume 1334 of *Lecture Notes in Computer Science*, pages 440–444. Springer-Verlag, 1997.
[37] C. Park, K. Itoh, and K. Kurosawa. Efficient anonymous channel and all/nothing election scheme. In T. Helleseth, editor, *Advances in Cryptology—EUROCRYPT 93*, volume 765 of *Lecture Notes in Computer Science*, pages 248–259. Springer-Verlag, 1994, 23–27 May 1993.
[38] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer-Verlag, 1992, 11–15 Aug. 1991.
[39] A. Pfitzmann, B. Pfitzmann, and M. Waidner. Isdn-mixes: Untraceable communication with small bandwidth overhead. In W. Effelsberg, H. W. Meuer, and G. Müller, editors, *Kommunikation in Verteilten Systemen*, volume 267 of *Informatik-Fachberichte*, pages 451–463. Springer, 1991.
[40] B. Pfitzmann. Breaking an efficient anonymous channel. In *EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 332–340, 1995.
[41] M. K. Reiter and A. D. Rubin. Crowds: anonymity for web transactions. *ACM Trans. Inf. Syst. Secur.*, 1(1):66–92, November 1998.
[42] A. D. Santis, S. Micali, and G. Persiano. Non-interactive zero-knowledge proof systems. In C. Pomerance, editor, *CRYPTO*, volume 293 of *Lecture Notes in Computer Science*, pages 52–72. Springer, 1987.
[43] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In R. Dingledine and P. Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop – PET ' 02*, volume 2482 of *Lecture Notes in Computer Science*. Springer-Verlag, April 2002.
[44] C. E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. J.*, 27:623–656, 1948.
[45] V. Shoup. Sequences of games: a tool for taming complexity in security proofs. cryptology eprint archive, report 2004/332, 2004.
[46] F. Stajano and R. J. Anderson. The cocaine auction protocol: On the power of anonymous broadcast. In A. Pfitzmann, editor, *Information Hiding*, volume 1768 of *Lecture Notes in Computer Science*, pages 434–447. Springer, 1999.
[47] B. Timmerman. A security model for dynamic adaptive traffic masking. In *NSPW '97: Proceedings of the 1997 workshop on New security paradigms*, pages 107–116, New York, NY, USA, 1997. ACM.
[48] D. Wikström. A universally composable mix-net. In M. Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 317–335. Springer, 2004.
[49] A. C. Yao. Protocols for secure computations. In *SFCS '82: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, pages 160–164, Washington, DC, USA, 1982. IEEE Computer Society.
[50] L. Zhuang, F. Zhou, B. Y. Zhao, and A. I. T. Rowstron. Cashmere: Resilient anonymous routing. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, pages 301–314. USENIX Association, 2005.

APPENDIX A.
GOLLE-JUELS SHORT DC-NET PROTOCOL

In this section, we fully specify Golle-Juels DC-net protocol using the language pWHILE. This protocol consists of at most 5 stages: an initialization stage I, transmission stage Tr, reception stage Col, computation stage Comp, and an optional reconstruction stage Rec. Let $k, \tau \in \mathbf{N}$, $\tau > 1$, be parameters and $[n]$ denote the set $\{1, \ldots, n\}$. For simplicity of the description, we assume the following programs (commands) are available. See [26] and references therein for their implementation.

1) $\mathsf{setupParam}(k)$: on input the security parameter k, creates system parameters, namely the admissible pairing e, descriptions of groups G_1, G_2, G, generators g, h of group G, and a hash function $H : \{0,1\}^* \to G_1$.
2) $\mathsf{polynomialInterp}(a_1, \ldots, a_n)$: returns $f(0)$ where $f(x)$ is unique t-degree polynomial such that $f(i) = a_i$ for $i \in [n]$.
3) $\mathsf{LagrangeCoef}(X, s)$: returns the Lagrange coefficients used for threshold cryptography.
4) $\mathsf{broadcast}(d)$: initiates a (reliable) broadcast protocol to transmit d.
5) $\mathsf{receive}()$: returns the value delivered by a (reliable) broadcast protocol initiated with $\mathsf{broadcast}(\cdot)$.
6) $\mathsf{element}(X)$: returns the lexicographically smallest element from set X.

Furthermore, during initialization phase, we assume the availability of a trusted third party $T \notin \{P_1, \ldots, P_n\}$ to privately compute all secret shares, all values $c_i = \pi(i)$ where π is randomly chosen secret permutation (i.e. to implement slot allocation [26]). The assumption on T can be removed by instantiating the protocol to n sequential executions of

the distributed protocol in [23], and using secure function evaluation [49] to compute the slots $c_1, \ldots, c_n$.[3]

GOLLE-JUELS PROTOCOL:

$$\begin{aligned}\mathrm{GJ}(M) \doteq\ & I; \mathrm{Tr}_1(M_1); \ldots ; \mathrm{Tr}_n(M_n);\\ & \mathrm{Col}_1; \ldots ; \mathrm{Col}_n;\\ & \mathrm{TRec}_1; \ldots ; \mathrm{TRec}_n; \mathrm{Rec}_1; \ldots ; \mathrm{Rec}_n;\\ & \mathrm{Comp}_1; \ldots ; \mathrm{Comp}_n;\end{aligned}$$

INITIALIZATION PHASE:

$$\begin{aligned}I =\ & g, h, e, G_1, G_2, G, H := \mathsf{setupParam}(k);\\ & \pi \xleftarrow{\$} \{\, p \,:\, \text{p is a permutation on } \{1, \ldots, n\} \,\};\\ & i := 0;\\ & \mathsf{while}\ i \le n\ \mathsf{do}\ \{\\ & \quad \mathsf{while}\ j \le n\ \mathsf{do}\\ & \quad \{x_{i,j} \xleftarrow{\$} \{1, \ldots, |G_2|\};\ y_{i,j} := g^{x_{i,j}};\ \}\\ & \quad x_i \xleftarrow{\$} \mathsf{polynomialInterp}(x_{1,i}, \ldots, x_{n,i});\\ & \quad y_i := g^{x_i};\ c_i := \pi(i);\}\\ & s := 0;\ \sigma := K;\end{aligned}$$

TRANSMISSION PHASE:

$$\begin{aligned}\mathrm{Tr}_i \doteq\ & \ell := 1; r := 0;\\ & \mathsf{while}\ \ell \le \tau\ \mathsf{do}\ \{\\ & \quad q_\ell := H(s||\ell);\\ & \quad \mathsf{if}\ i = 1\ \mathsf{then}\ j := 2\ \mathsf{else}\ j := 1; w_i[\ell] := 1;\\ & \quad \mathsf{while}\ j \le n\ \mathsf{do}\ \{\\ & \qquad \mathsf{if}\ i < j\ \mathsf{then}\ w_i[\ell] := w_i[\ell] \cdot e(q_\ell, y_j)^{x_i}\\ & \qquad\qquad \mathsf{else}\ w_i[\ell] := w_i[\ell] \cdot e(q_\ell, y_j)^{-x_i}\\ & \qquad j := j + 1; \mathsf{if}\ j = i\ \mathsf{then}\ j := j + 1;\}\\ & \quad r_i[\ell] \xleftarrow{\$} \mathbb{Z}_p; r := r + r_i[\ell];\\ & \quad \mathsf{if}\ \ell = c_i\\ & \qquad \mathsf{then}\ v_i[\ell] := M_i \cdot w_i[\ell]; d_i[\ell] := g \cdot h^{r_i[\ell]}\\ & \qquad \mathsf{else}\ v_i[\ell] := w_i[\ell]; d_i[\ell] := h^{r_i[\ell]};\\ & \quad \sigma_i[\ell] \xleftarrow{\$} P^1_{\mathtt{gen}}(x_i, r_i[\ell]; v_i[\ell], \vec{y}, q_\ell, d_i[\ell], g, h);\\ & \quad \ell := \ell + 1;\}\\ & \mu_i \xleftarrow{\$} P^2_{\mathtt{gen}}(r; d_i[1], \ldots, d_i[n], g, h);\\ & \mathsf{broadcast}(v_i, d_i, \sigma_i, \mu_i);\end{aligned}$$

RECEPTION PHASE:

$$\begin{aligned}\mathrm{Col}_i \doteq\ & (v_1, d_1, \sigma_1, \mu_1), \ldots, (v_n, d_n, \sigma_n, \mu_n) := \mathsf{receive}();\\ & j := 1; \Delta := \emptyset;\\ & \mathsf{while}\ j \le n\ \mathsf{do}\ \{\\ & \quad \alpha_j := V^2(\mu_j, ; d_j[1], \ldots, d_j[\tau], g, h); \ell := 1;\\ & \quad \mathsf{while}\ \ell \le \tau\ \mathsf{do}\ \{\\ & \qquad \beta_j[\ell] := V^1(\sigma_j[\ell], v_j[\ell], \vec{y}, d_j[\ell], q_\ell, g, h);\\ & \qquad \mathsf{if}\ \alpha_j = 0\ \mathsf{or}\ \beta_j[\ell] = 0\ \mathsf{then}\ \Delta := \Delta \cup \{j\};\\ & \qquad l := l + 1\}\\ & \quad j := j + 1\}\end{aligned}$$

RECONSTRUCTION PHASE:

$$\begin{aligned}\mathrm{Rec}_i \doteq\ & \Delta' := \Delta;\\ & \mathsf{while}\ \Delta \neq \emptyset\ \mathsf{do}\ \{\\ & \quad j := \mathsf{element}(\Delta); \Delta := \Delta \setminus \{j\}; \ell := 1;\\ & \quad \mathsf{while}\ \ell \le \tau\ \mathsf{do}\ \{\\ & \qquad \Delta_{j,i}[\ell] := q_\ell^{x_{j,i}}; \mathsf{broadcast}(\Delta_{j,i}[\ell]);\\ & \qquad \ell := \ell + 1;\}\}\end{aligned}$$

$$\begin{aligned}\mathrm{TRec}_i \doteq\ & \Delta' := \Delta;\\ & \mathcal{B} := \{1, \ldots, n\} \setminus \Delta'; \Gamma := \mathcal{B};\\ & \mathsf{while}\ \Delta \neq \emptyset\ \mathsf{do}\ \{\\ & j := \mathsf{element}(\Delta); \Delta := \Delta \setminus \{j\}; \ell := 1;\\ & \mathsf{while}\ \ell \le \tau\ \mathsf{do}\ \{\\ & \Delta_{j,1}[\ell], \ldots, \Delta_{j,n}[\ell] := \mathsf{receive}();\\ & \mathcal{B} := \Gamma;\\ & \mathsf{while}\ \mathcal{B} \neq \emptyset\ \mathsf{do}\ \{\\ & j' := \mathsf{element}(\mathcal{B}); \mathcal{B} := \mathcal{B} \setminus \{j'\};\\ & \mathsf{if}\ e(\Delta_{j,j'}[\ell], g) \neq e(y_{j,j'}, q_\ell)\ \mathsf{then}\ \Gamma := \Gamma \setminus \{j'\};\}\\ & \ell := \ell + 1;\}\}\\ & \mathsf{if}\ |\Gamma| < m - n\ \mathsf{then}\ \mathsf{abort};\\ & \lambda_1, \ldots, \lambda_n := \mathsf{LagrangeCoef}(\Gamma, m - n); \Delta := \Delta';\\ & \mathsf{while}\ \Delta \neq \emptyset\ \mathsf{do}\ \{\\ & j := \mathsf{element}(\Delta); \Delta := \Delta \setminus \{j\}; \ell := 1;\\ & \mathsf{while}\ \ell \le \tau\ \mathsf{do}\ \{\\ & z'_j[\ell] := 1; \Gamma' := \Gamma;\\ & \mathsf{while}\ \Gamma \neq \emptyset\ \mathsf{do}\ \{\\ & \quad j' := \mathsf{element}(\Gamma); \Gamma := \Gamma \setminus \{j'\};\\ & \quad z'_j[\ell] := z'_j[\ell] \cdot \lambda_{j'} \cdot \Delta_{j,j'}[\ell];\}\\ & v_j := 1; \mathsf{if}\ j = 1\ \mathsf{then}\ u := 2\ \mathsf{else}\ u := 1;\\ & \mathsf{while}\ u \le n\ \mathsf{do}\ \{\\ & \mathsf{if}\ j < u\\ & \quad \mathsf{then}\ v_j[\ell] := v_j[\ell] \cdot e(z'_j[\ell], y_u)\\ & \quad \mathsf{else}\ v_j[\ell] := v_j[\ell] \cdot e(z'_j[\ell], y_u)^{-1};\\ & u := u + 1; \mathsf{if}\ u = j\ \mathsf{then}\ u := u + 1\};\\ & \ell := \ell + 1;\}\}\end{aligned}$$

MESSAGE COMPUTATION PHASE:

$$\begin{aligned}\mathrm{Comp}_i \doteq\ & \ell := 1;\\ & \mathsf{while}\ \ell \le \tau\ \mathsf{do}\ \{\\ & \quad j := 1;\\ & \quad \mathsf{while}\ j \le n\ \mathsf{do}\ \{\\ & \qquad s_i[\ell] := s_i[\ell] \cdot v_j[\ell]; j := j + 1;\}\\ & \quad \ell := \ell + 1;\}\end{aligned}$$

[3]We chose to perform slot allocation using (possibly inefficient) multiparty computation techniques as it is arguably the simplest scenario under which Golle-Juels' interference and communication robustness can be analyzed. Alternative (or none) slot reservation mechanisms [26] could also be used, which clearly yields protocols with different values for interference/communication robustness.

APPENDIX B.
ZERO KNOWLEDGE PROOF OF KNOWLEDGE

The following description is adapted from [2]. Let $\eta \in \mathbf{N}$ be a security parameter, and $p_1, \ldots, p_5$ fixed polynomials. We define an input domain $D_{\mathtt{in}} = \{\,0,1\,\}^{\eta}$, a common reference string domain $D_{\mathtt{crs}} = \{\,0,1\,\}^{p_1(\eta)}$, a witness domain $D_{\mathtt{wit}} = \{\,0,1\,\}^{p_2(\eta)}$, a proof domain $D_{\mathtt{pf}} = \{\,0,1\,\}^{p_3(\eta)}$, a simulation trapdoor domain $D_{\mathtt{sim}} = \{\,0,1\,\}^{p_4(\eta)}$, and an extraction trapdoor domain $D_{\mathtt{ext}} = \{\,0,1\,\}^{p_5(\eta)}$.

Definition 5 (Non-Interactive Zero-Knowledge, aka NIZK): A NIZK proof system (or protocol) consists of a tuple $\mathcal{P} = (R, K, P_{\mathtt{gen}}, V, S, E)$ where

- The NP relation $R(x, w)$ ranges over $D_{\mathtt{in}} \times D_{\mathtt{wit}}$. We say w is a *witness of membership* of x.
- The *common reference string (CRS) generator* program K is a randomized program which takes as input the security parameter η (written in unary) and outputs a string $\sigma \in D_{\mathtt{crs}}$, called the common reference string (or CRS).
- The *proof generating* program $P_{\mathtt{gen}}$ is a randomized program which takes a CRS σ, an input $x \in D_{\mathtt{in}}$, and

a witness $w \in D_{\mathtt{wit}}$, and outputs a string $\pi \in D_{\mathtt{pf}}$, called a proof.

- The *proof verification* program V is a randomized program which takes a CRS σ, an input $x \in D_{\mathtt{in}}$, and a proof $\pi \in D_{\mathtt{pf}}$, and outputs a bit b. If $b = 1$ we say σ is a valid proof for x under CRS σ.
- The *simulator* $S = (S_1, S_2)$ is a pair of randomized programs. Algorithm S_1 on input the security parameter η (in unary) outputs a simulated CRS $\sigma' \in D_{\mathtt{crs}}$ and a simulation trapdoor $\tau \in D_{\mathtt{sim}}$. Algorithm S_2 on input σ', τ, and a string $x \in D_{\mathtt{in}}$, outputs a simulated proof $\pi' \in D_{\mathtt{pf}}$.
- The *knowledge extractor* $E = (E_1, E_2)$ is a pair of possibly randomized programs. Algorithm E_1 on input η (the security parameter in unary) outputs a simulated CRS $\sigma' \in D_{\mathtt{crs}}$ and an extraction trapdoor $\xi \in D_{\mathtt{ext}}$. Algorithm E_2 on input σ', ξ, $x \in D_{\mathtt{in}}$, and a proof $\pi \in D_{\mathtt{pf}}$ outputs a string $w' \in D_{\mathtt{wit}}$.

Definition 6 (Non-Interactive Zero-Knowledge Proof system): A proof system $\mathcal{P} = (R, K, P_{\mathtt{gen}}, V, R, S, E)$ is said to be an extraction zero-knowledge proof system if it satisfies the following four properties:

1) **(Completeness)**: For every polynomial-time algorithm A, the probability $|\Pr[\mathtt{ZKCOMP};\ b' = 1]|$ for the command

$$\begin{aligned}\mathtt{ZKCOMP} \doteq\ & \sigma := K;\ (x, w) := A(\sigma);\\ & \pi := P_{\mathtt{gen}}(\sigma, x, w);\\ & \mathsf{if}\ (x, w) \notin R \text{ or } V(\sigma, x, \pi) = 1\\ & \quad \mathsf{then}\ b' := 0\\ & \quad \mathsf{else}\ b' := 1\end{aligned}$$

is negligible on the security parameter η.

2) **(Proof of Knowledge)**: Consider the following commands

$$\begin{aligned}\mathtt{INDCRSE} \doteq\ & b \xleftarrow{\$} \{0, 1\};\\ & \mathsf{if}\ b = 1\ \mathsf{then}\ \sigma := K\ \mathsf{else}\ (\sigma, \xi) := E_1;\\ & b' := A_0(\sigma);\end{aligned}$$

and

$$\begin{aligned}P_{\mathtt{extr}} \doteq\ & (\sigma, \xi) := E_1; (x, \pi) := A_1(\sigma);\\ & w := E_2(\sigma, \xi, x, \pi);\\ & \mathsf{if}\ (x, w) \notin R \text{ and } V(\sigma, x, \pi) = 1\\ & \quad \mathsf{then}\ b := 1\\ & \quad \mathsf{else}\ b := 0\end{aligned}$$

Then, for any polynomial-time programs A_0, A_1 the probabilities $|\Pr[\mathtt{INDCRSE};\ b' = b] - \frac{1}{2}|$ and $\Pr[P_{\mathtt{extr}};\ b = 1]$ are negligible in η.

3) **(Zero-Knowledge)**: Consider the commands

$$\begin{aligned}\mathtt{INDCRSS} \doteq\ & b \xleftarrow{\$} \{0, 1\};\\ & \mathsf{if}\ b = 1\ \mathsf{then}\ \sigma := K\ \mathsf{else}\ (\sigma, \tau) := S_1;\\ & b' := A_0(\sigma);\end{aligned}$$

and

$$\begin{aligned}\mathtt{ZK} \doteq\ & (\sigma, \tau) := S_1;\ (x, w, s) := A_1(\sigma);\ b \xleftarrow{\$} \{0, 1\};\\ & \mathsf{if}\ b = 1\\ & \quad \mathsf{then}\ \pi := P_{\mathtt{gen}}(\sigma, x, w)\\ & \quad \mathsf{else}\ \pi := S_2(\sigma, \tau, x);\\ & b' := A_2(\pi, s);\end{aligned}$$

Then, the probabilities $|\Pr[\mathtt{INDCRSS};\ b' = b] - \frac{1}{2}|$ and $|\Pr[\mathtt{ZK};\ b' = b \wedge (x, w) \in R] - \frac{1}{2}|$ are negligible on η for any polynomial-time adversaries A_0, A_1 and A_2, where A_0 reads σ and writes a bit b', A_1 reads σ and writes x, w, s, and A_2 reads π, s and writes b'.

APPENDIX C. ROBUSTNESS PROOF

Proof: (of Theorem 5.1)

The proof of the theorem uses standard game-hopping technique. We write the intermediary games using the pWHILE language, and as such it may by somewhat unfamiliar to cryptographers. We choose to present the games (and the transformations) in this language for simplicity, and note that the difference from standard cryptographic proofs is only superficial. We explain in some of the instances how our games/transformations correspond to the more standard ones.

For this proof we assume that there are n participants where $n - t$ are honest and $n - t > \frac{n}{2}$. For brevity, let $m = n - t$.

We assume that the protocol uses an extraction zero-knowledge proof system [2] satisfying completeness, proof of knowledge, and zero-knowledge as defined in Definition 6. We also assume the protocol uses a bilinear pairing scheme that satisfies the BDDH[4] assumption.

We also use a $(n-t)$-out-of-n threshold (verifiable) secret-sharing scheme [38].

We model the protocol with $n - t$ honest participants in our language with the phases detailed in Appendix A:

$$\begin{aligned}& GJ[A_1, A_2](M) \doteq\\ & I;\\ & \mathrm{Tr}_1(M_1); \ldots; \mathrm{Tr}_m(M_m);\\ & (v_{m+1}, d_{m+1}, \sigma_{m+1}, \mu_{m+1}), \ldots, (v_n, d_n, \sigma_n, \mu_n)\\ & \qquad \leftarrow A_1((v_1, d_1, \sigma_1, \mu_1), \ldots, (v_m, d_m, \sigma_m, \mu_m))\\ & \mathrm{Col}_1; \ldots; \mathrm{Col}_m;\\ & \mathrm{TRec}_1; \ldots; \mathrm{TRec}_m;\\ & \Delta_{m+1}, \ldots \Delta_n \leftarrow A_2(\Delta_1, \ldots \Delta_m)\\ & \mathrm{Rec}_1; \ldots; \mathrm{Rec}_m;\\ & S_1 \leftarrow \mathrm{Comp}_1; \ldots; S_m \leftarrow \mathrm{Comp}_m;\end{aligned}$$

Code with subindexes from 1 to m represents the protocol code of the honest participants (as given in Appendix A), programs A_1 and A_2 represent code from the adversary

[4]Bilinear Decisional Diffie-Hellman

that as usual is unknown and polynomially bounded (on the security parameter). The code:

$$\begin{array}{ll}\mathsf{CR} \doteq & M^0, N \leftarrow A(); \\ & \vec{M} = M^0 + +N; \\ & \vec{S} \leftarrow GJ[A_1, A_2](M^0) \\ & \mathsf{if}\ |M^0 \cap \vec{S}| < 2m - n\ \mathsf{then}\ b' := 1\ \mathsf{else}\ b' := 0\end{array}$$

then expresses the security game for $(2m - n)$-communication robustness of the GJ protocol.

Let $\vec{S}$ be the set of received messages at the end of the protocol, that is output by any of the participants. Notice that the set S_i of received messages by participant i (there is one S_i for each participant in $\vec{S}$) is the same for all i from 1 to m due to use of the broadcast channel.

Notice that the adversary possesses the private keys of the t (corrupt) participants.

After the transmission phase for the honest participants, the adversary transmits vectors with corresponding proofs of knowledge of the t participants that he corrupts. Since we assume that the adversary does not follow the protocol at some point (otherwise we can consider he is an honest participant, and the proof becomes uninteresting), then he can be in one of the two following cases:

1) The adversary transmits a vector v and proof σ such that there is a position ℓ in the vector where the verification fails, that is $V^1(\sigma[\ell], v[\ell], y_1, \ldots, y_n, d[\ell], q_l, g, h) = 0$. (The case where the proof μ_i is incorrect is analogous to the case where σ_i and omitted.)
2) The adversary transmits a vector v and proof σ such that the verification succeeds, that is for all ℓ, we have $V^1(\sigma[\ell], v[\ell], y_1, \ldots, y_n, d[\ell], q_l, g, h) = 1$. In this case we assume that either the vector v is incorrect (the vector does not satisfies the format of containing $n-1$ slots filled with the padding) or the adversary does not respect the slot assignment protocol by changing the value of c_i, and produces a collision with an honest participant. (Notice that it is easy for a rushing adversary knowing the private keys of all dishonest participant and vectors of the honest participants, to calculate a vector that will collide with the vector of an honest participant: it suffices that he calculates the padding of the dishonest participants, and multiply it with the vector of all honest participants. As a result he will obtain a vector with all messages and positions where the honest participants transmit. Then he just transmits a vector with a message in some of these positions.)

Assume that there are k_1 ($0 \leq k_1 \leq t$) vectors of adversary A_1 that lie in the first case and k_2 ($0 \leq k_2 \leq t$) vectors of adversary A_1 that lie in the second case (we assume $k_1 + k_2 = t$). For simplicity in the description, we assume that vectors in the first case are the first vectors from vector $m + 1$ up to vector $m + k_1$. The proof proceeds as follows:

For the k_1 vectors lying in case 1, we use the hypotheses on program semantics, and security of the threshold sharing scheme to conclude that we can recover the private keys of the (k_1) dishonest participants, and thus reconstruct their paddings. We do not present the full reduction associated to this step.

We can therefore conclude that $GJ[A_1, A_2](M^0)$ is semantically equivalent (in terms of the distribution of the $\vec{S}$ vector for the honest participants, that is the distribution of $|M^0 \cap \vec{S}|$ in CR) to a program GJ_1 where honest code without transmitting any message (input is a message that is neutral for the appropriate group multiplication):

$$\begin{array}{l}GJ_1[A_1, A_2](M) \doteq \\ I; \\ \mathrm{Tr}_1(M_1); \ldots; \mathrm{Tr}_m(M_m); \\ \mathrm{Tr}_{m+1}(1); \ldots; \mathrm{Tr}_{m+k_1}(1); \\ (v_{m+k_1+1}, d_{m+1}, \sigma_{m+1}, \mu_{m+1}), \ldots, (v_n, d_n, \sigma_n, \mu_n) \\ \qquad \leftarrow A_1((v_1, d_1, \sigma_1, \mu_1), \ldots, (v_m, d_m, \sigma_m, \mu_m) \\ \mathrm{Col}_1; \ldots; \mathrm{Col}_m; \\ \mathrm{TRec}_1; \ldots; \mathrm{TRec}_m; \\ \Delta_{m+1}, \ldots \Delta_n \leftarrow A_2(\Delta_1, \ldots \Delta_m) \\ \mathrm{Rec}_1; \ldots; \mathrm{Rec}_m; \\ S_1 \leftarrow \mathrm{Comp}_1; \ldots; S_m \leftarrow \mathrm{Comp}_m;\end{array}$$

For the other vectors of the adversary (vectors whose proof of knowledge do verify) we use the hypotheses on program semantics, and proof of knowledge.

We use the property of proof of knowledge to show that these vectors actually satisfy the property of containing at most one position with something different of a padding (the adversary cannot cheat for a proof that verifies).

We first apply the game `INDCRSE` of proof of knowledge. We obtain $GJ_2[A_1, A_2](M^0)$ by replacing in the initialization phase I of $GJ_1[A_1, A_2](M^0)$, the common random string generator K by E_1. The distributions generated by the game CR in which $GJ_1[A_1, A_2]$ is replaced with $GJ_2[A_1, A_2]$ can only be distinguished with negligible probability $\epsilon_0(\nu)$. Otherwise, we construct an adversary that breaks the `INDCRSE` property. Notice that this is a standard game hopping step that is based on the security of the zero-knowledge proof of knowledge protocol. Assume that $GJ_3[A_1, A_2](M^0)$ is as $GJ_2[A_1, A_2](M^0)$ except that we insert new variables $w_{m+k_2+1}, \ldots w_t$ that are assigned with $E_2(\sigma, v_i, d_i, \sigma_i, \mu_i)$ with $i \in \{1 \ldots t\}$. Since this is deadcode for the computation of $\vec{S}$, the distributions are equivalent.

We apply the game P_{extr} of proof of knowledge to conclude that the k_2 vectors generated by the adversary that verify should satisfy (with overwhelming probability) that

they contain a correct padding in all positions except for one position.

Let $GJ_4[A_1, A_2]$ be as $GJ_3[A_1, A_2]$ except that we unfold the while of the Col_i phase and replace the verification for the proof of knowledge of a vector $(v_i, d_i, \sigma_i, \mu_i)$ of the adversary by a conjunction of the verification of the proof of knowledge and the assertion that $(v_i, d_i, \sigma_i, \mu_i)$ and the witness w_i given by the extractor are in a relation that imply that v_i contains correct paddings except for maybe one position. The distance between $GJ_4[A_1, A_2]$ and $GJ_3[A_1, A_2]$ can only be distinguished with negligible probability $\epsilon_1(\nu)$.

Moreover, since vectors transmitted by the adversary hold correct proofs of knowledge, then there are no necessity of reconstruction phase. So we can show by semantics that $GJ_4[A'_1, A_2](M^0)$ is equivalent to the program where there is no reconstruction phase:

$$
\begin{array}{l}
GJ_5[A'_1, A_2](\vec{M}) \doteq \\
I; \\
\mathrm{Tr}_1(M_1); \ldots; \mathrm{Tr}_m(M_m); \\
\mathrm{Tr}_{m+1}(1); \ldots; \mathrm{Tr}_{m+k_1}(1); \\
(v_{m+k_1+1}, d_{m+k_1+1}, \sigma_{m+k_1+1}, \mu_{m+k_1+1}), \ldots, (v_n, d_n, \sigma_n, \mu_n) \\
\qquad \leftarrow A_1((v_1, d_1, \sigma_1, \mu_1), \ldots, (v_m, d_m, \sigma_m, \mu_m) \\
\mathrm{Col}_1; \ldots; \mathrm{Col}_m; \\
S_1 \leftarrow \mathrm{Comp}_1; \ldots; S_m \leftarrow \mathrm{Comp}_m;
\end{array}
$$

Finally by semantics, and because vectors transmitted with a proof of knowledge must obey the property of transmitting a unique message par participant (except for negligible probability), we have that the adversary can only produce k_2 collisions. Since k_2 is at most equal to t we obtain that in:

$$
\begin{array}{ll}
\mathsf{CR}_5 \doteq & M^0, N \leftarrow A(); \\
& \vec{M} = M^0 + +N; \\
& S \leftarrow GJ_5[A'_1, A_2](M^0) \\
& \mathsf{if}\ |M^0 \cap S| < 2m - n\ \mathsf{then}\ b' := 1\ \mathsf{else}\ b' := 0
\end{array}
$$

exactly $n - (k_1 + k_2)$ messages are received, which is at least $n - t$ are received with overwhelming probability. The adversary wins CR_5 with negligible probability $\epsilon_2(\nu)$ where the probability $\epsilon_2(\nu)$ is $Pr[\mathsf{CR}_5; b' = 1] = Pr[\mathsf{CR}_4; b' = 1]$ and $Pr[\mathsf{CR}_2; b' = 1] = Pr[\mathsf{CR}_3; b' = 1]$. Finally $Pr[\mathsf{CR}_1; b' = 1] = Pr[\mathsf{CR}; b' = 1]$. We conclude that the adversary wins CR with negligible probability since it is $\epsilon_2(\nu) + \epsilon_1(\nu) + \epsilon_0(\nu)$, the sum of negligible functions.

INTERFERENCE ROBUSTNESS: Since in the transform GJ_5 protocol, there are at most t vectors generated by the adversary, it is easy to see that the adversary can produced at most t collisions. Thus, t honest messages can be tampered (furthermore t messages in $\vec{M}$ are not delivered since they correspond to messages from the adversary), hence the bound for interference-robustness is $3t$ (t honest messages that are not included in $\vec{S}$ plus t messages that are in $\vec{S}$ but come from the adversary plus t messages that were in $\vec{M}$ but correspond to the corrupt participants).

■

Proof: (of Theorem 5.2) The modified protocol, replaces the transmission of the message in the transmission phase (see Appendix A) by the following:

$$
\begin{array}{l}
\mathsf{if}\ \ell = c_i \\
\qquad \mathsf{then}\ x \xleftarrow{\$} Z_q; v_i[\ell] := (M_i||x||H(M_i||x)) \cdot w_i[\ell] \\
\qquad \mathsf{else}\ v_i[\ell] := w_i[\ell];
\end{array}
$$

Furthermore, broadcasting protocols for transmission of vectors are replaced by simultaneous broadcasting (to avoid giving to the adversary any fresh value x before the adversary transmits his own vector). In the message computation phase (see Appendix A), before validating a message by putting it in $\vec{S}$ (vector $s[.]$), tags are verified. Only messages with valid tags are saved. The proof of this theorem is just an extension of the proof of Theorem 5.1. The proof of communication robustness for the modified version of short DC-nets is exactly the same. The proof of interference robustness is just a further step of the previous proof, where we can eliminate tampered messages in the message computation phase by hypotheses of tagging schemes the k_2 vectors produced by the adversary in GJ_5. In fact, by obliviously non-malleable randomized tagging scheme security, the adversary cannot generate a message M^* such that multiplying M^* by a message $M_i||x||H(M_i||x)$ of an honest participant it will return a correct tag t' for the multiplication of M^* and $M_i||x||H(M_i||x)$. Hence, the adversary cannot tamper any honest message without being detected. Since the protocol is $2t$-interference robustness (optimal interference robustness) since he can at least produced t collisions (messages that withdrawn from $\vec{S}$) and prevent t messages from the initial set $\vec{M}$ to be delivered (his own messages).

■